Multi-objective variable subset selection using heterogeneous surrogate modeling and Sequential Design

Joachim van der Herten, Ivo Couckuyt, Dirk Deschrijver and Tom Dhaene
Internet Based Communication Networks and Services (IBCN)
Ghent University - iMinds
Technologiepark 15, B-9052 Gent, Belgium
Email: joachim.vanderherten@intec.ugent.be

***Abstract*—Constructing surrogate models of high-dimensional complex black-box systems from simulation-based data requires an appropriate choice of surrogate model type, as well as identification of the most influential input parameters. As including irrelevant input parameters results in a longer surrogate model training process and potentially increases the risk of overfitting, it is important to identify a small set of relevant parameters during the adaptive modeling phase of the surrogate modeling process. A multi-objective optimization step is proposed to identify both the appropriate model type as well as a parameters subset. The obtained model can be used for evaluation intensive applications such as exploration, sensitivity analysis or optimization.**

I. Introduction

Confronted with high-fidelity simulation codes approximating complex systems to avoid countless (expensive) prototypes during product design, engineers nowadays approximate the simulator response with cheap-to-evaluate surrogate models (also known as response surface models or meta-models) [1]. These surrogate models are constructed by selecting a set of input values for the simulator and computing the responses. The surrogate models then learns the relation between the inputs and the simulator output and can be used to replace the simulator for applications requiring many evaluations such as domain exploration, sensitivity analysis, or (multi-objective) optimization of the underlying complex system it approximates. Several types of regression techniques are typically used as surrogate models including Artificial Neural Networks (ANN) [2], Kriging [3], Radial Basis Functions (RBF), Support Vector Machines (SVM) [4], [5], [6], Least-Squares Support Vector Machines (LS-SVM) [7], Extreme Learning Machines (ELM) [8], [9], Splines, Rational interpolation [10], etc. The most suitable model type depends on the properties of the response surface of the simulator [11]. For example, Kriging usually performs well for smooth responses, whereas ANN models are able to capture strong non-linear output behaviour.

The number of samples is kept small as each additional simulator evaluation is very expensive in terms of computing time. This implies each sample should contribute a maximal amount of information to improve the accuracy of the surrogate model. The set of samples can be chosen at once (one-shot design): all input combinations are chosen, simulated and a model is trained. As it is difficult to decide how much samples are needed for an accurate surrogate model upfront, this approach involves a risk of evaluating too few or too many samples. To avoid this, sequential design has become a popular method [12]. At the start of the process a small set of data points is evaluated and followed by the *adaptive modeling* step to create an intermediate model. The parameters of the models (*hyperparameters*) are optimized automatically with respect to a quality estimator specified upfront. Next, an adaptive sampling algorithm such as FLOLA-Voronoi [13], [14] ,Efficient Global Optimization (EGO) [15], or Efficient Multi-objective Optimization (EMO) [16] selects new data points (possibly based on knowledge obtained during the evaluation and modeling phase). After evaluation these samples are added to the data set and the adaptive modeling step is repeated to create an updated intermediate model. This process continues iteratively until the modeling goals have been met, or any limitation is reached (maximum number of data points, or a time-limit). The process of surrogate modeling with sequential design is illustrated in Fig. 1.

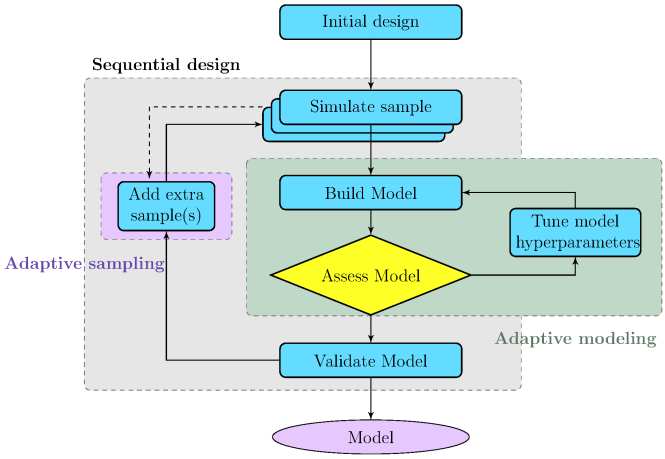


Fig. 1. Schematic illustration of the surrogate modeling process with sequential design.

An important aspect of surrogate modeling is the assumption

of a black-box simulator. No a priori information is available on the response behaviour of the simulator: all information is learned from the simulator evaluations. This restriction has two important consequences:

- Picking the optimal surrogate model type is difficult. Especially when only a small number of data points is available (as is often the case for sequential design) it is difficult to estimate the properties of the response surface, and choose the appropriate model type. Often this is guessed based on earlier experiments or experience of the product designers. Earlier, an evolutionary method for model type selection during adaptive modeling was introduced to automate this choice [17].
- Because the relation between the input and output is unknown, some input parameters may influence the output more than others. In fact, some parameters can even be irrelevant. Including these parameters within the modeling potentially makes modeling results worse, as overfitting may occur. This is one of the motivations for feature selection techniques in the field of machine learning. Therefore automatic identification of the most significant parameters during the adaptive modeling step can significantly enhance the model quality for high-dimensional surrogate modeling problems.

This paper presents an automatic optimization procedure which aims to find both the optimal subset of input parameters and the ideal surrogate modeling type by considering the model selection criterion as a multi-objective optimization problem. It is an extension of earlier work on automatic surrogate model type selection presented in [17], as well as principles presented in [18]. Section II introduces the idea of adaptive global surrogate modeling formally. An overview of the motivation and brief overview of the three main types of feature selection is given in Section III. The proposed approach based on multi-objective optimization is given in Section IV, and illustrated on a 30-dimensional example in Section V.

II. Adaptive Global Surrogate Modeling

The process of constructing a surrogate model can be mathematically expressed as follows: given an unknown function $f : \Omega \to \mathbb{C}^p$ defined over the input domain $\Omega \subset \mathbb{R}^d$, whose function values $Y = \{f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N)\}$ are known for a set of distinct query points $X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$. Together, they form the data set $D = \{(\boldsymbol{x}_1, f(\boldsymbol{x}_1)), ...(\boldsymbol{x}_N, f(\boldsymbol{x}_N))\} \subset \mathbb{R}^d$. A suitable function $\tilde{f}$ from an approximation space S with $\tilde{f} : \Omega \to \mathbb{C}^p \in S$ has to be chosen based on some criterion ξ. This criterion consists of 3 different aspects:

$$\xi = (\Lambda, \epsilon, \tau) \tag{1}$$

Given $\epsilon \in E$, an error function from the set of error functions and τ the target error specified by the user, Λ is a quality estimator with $\Lambda : E \times \mathcal{S} \times \mathcal{P}(\Omega) \to \mathbb{R}^+$ (usually higher quality means lower errors), and τ the target error specified by the user. Finding the optimal approximation function $\tilde{f}^* \in S$ can now be expressed as

$$\begin{aligned} &\arg\min_{t \in T} \ \arg\min_{\theta \in \Theta} && \Lambda(\epsilon, \tilde{f}_{t,\theta}, D) \\ &\text{subject to} && \Lambda(\epsilon, \tilde{f}_{t,\theta}, D) \leq \tau. \end{aligned} \tag{2}$$

with $\tilde{f}_{t,\theta}$ the surrogate model of type t with model parameters θ (from a parameter space Θ, specific for each surrogate model type).

The first minimization handles the model type selection aspect of the surrogate modeling process, whereas the second optimization problem tunes the *hyperparameters* of the model to find an optimum of the model quality estimator Λ. The choice of Λ is therefore crucial to obtain a satisfying surrogate model at the end of the process. Consulting the users of the surrogate model and defining what is expected from the model and what is not, is a good starting point. These requirements can be formally translated into a good quality estimator.

A straightforward approach is minimizing the error between the surrogate modeling response and the true responses Y for the training input data X. This is often referred to as *training error* or *sample error* and pushes the hyperparameter optimization to models interpolating the data points perfectly. This rarely provides a satisfying model as the optimization problems do not consider model quality in $\Omega \setminus X$, which will lead to very unreliable responses when new data points $\boldsymbol{x} \notin X$ are to be predicted (poor *generalization performance*). A popular method to enhance generalization performance is crossvalidation. Other model quality estimators include Akaike Information Criterion (AIC) [19], Linear Reference Model (LRM) [12], jack-knife, Validation sets, etc. Λ may also be composed as linear combination of several quality estimators, or the optimization problem of Equation 2 can be optimized multi-objectively [20]. For the latter case not a single $\tilde{f}^* \in S$ is found, but a pareto-optimal set $P \subset S$ representing the trade-off between the quality estimators.

III. Feature Selection

Traditionally, adaptive surrogate modeling includes all d-dimensions of the input space Ω in the process. When the dimensionality of the input space grows, the process becomes more difficult as several issues start to occur. Training some model types for high-dimensional problems is infeasible: for instance Kriging with the anisotropic Gaussian correlation function has a hyperparameter for each dimension which needs to be optimized, which turns the inner optimization of Equation 2 into a complex high-dimensional optimization problem. Furthermore, the size of the input space grows exponentially in terms of d: this implies more data points are required to capture all dynamics of the response surface, and to reduce the uncertainty of the model. For the Hölder class of functions the lower bound on the error τ was formally proven [21]:

$$\tau \geq cN^{-\frac{k+\alpha}{d}},$$

for a constant c, and parameters k and α representing the smoothness of the function. While not completely representative

of the surrogate modeling scenario, it does give a (intuitive) hint to the scalability issues associated with it. If the function is more smooth, the lower bound for the error drops faster as more samples are evaluated, whereas increasing dimensionality makes the drop less significant. Evaluating the simulator for many data points is already a computationally demanding step, but it also increases the computational burden of model training.

Fortunately, usually only a few input parameters have a significant influence on the response behaviour of the simulator. Other parameters only contribute very little, or are completely irrelevant. Automatically identifying the group of relevant features is referred to as *Feature Selection* and aims to achieve 3 goals [22], [23]:

- Simpler models are easier to interpret and provide useful information on the parameter relevance.
- Shorter model training time.
- Improve generalization by reducing overfitting risk: including irrelevant parameters in the model training increases the risk of overfitting, especially when only a relatively small amount of data is available with respect to the dimensionality.

Three flavours of feature selection methods exist: *filter* only analyses the data set D, and does not require any type of models. *Wrapper* methods build a model of the data and analyse it to identify feature relevance. *Embedded* methods handle the feature selection during the training phase of the model. As small data sets are usually encountered (especially in the context of sequential design) it may be dangerous to apply filter methods as not all information is available (yet). Furthermore, the overfitting risk due to the small number of observations in combination with the dimensionality of the input space may influence the result. Instead, we propose to build many models (of different types) during the adaptive modeling phase, on different subsets of input variables. By incorporating the feature selection into the adaptive modeling phase of the surrogate modeling process, information on the model performance can be used to guide the selection process and find an optimal input-subspace. More specifically, this paper introduces an approach by incorporating feature selection into the model quality estimator. This optimization is driven by a multi-objective Genetic Algorithm (GA), similar to approaches suggested in [24], [25]. This means the proposed approach is mostly a wrapper method.

IV. Proposed multi-objective search strategy

To incorporate feature selection as a part of the surrogate modeling process, the model quality estimator Λ is expanded to penalize *complex* models, in addition to a component recording the model error the function outputs a complexity objective. Formally this extended model quality estimator can be written as

$$\begin{aligned}\tilde{\Lambda}: \quad E \times \mathcal{S}^\star \times \mathcal{P}(\Omega^\star) &\rightarrow (\mathbb{R}^+, \mathbb{R}^+) \\ (\epsilon, \tilde{f}_{\lambda,\theta}, D_z) &\mapsto \left(\Lambda(\epsilon, \tilde{f}_{\lambda,\theta}, D_z), \mathcal{C}(\tilde{f}_{\lambda,\theta})\right). \quad (3)\end{aligned}$$

The approximation function now has a parameter $\lambda = (t, z)$, with z a vector defining the included input variables. Therefore, D_z is a lower-dimensional subspace of D spanned by the variables in z. Because of this projection $\tilde{f}$ is taken from a different approximation space $\mathcal{S}^\star$ and $\tilde{f}_{\lambda,\theta} : \Omega^\star \rightarrow \mathbb{C}^p$ with $\Omega^\star \subset \mathbb{R}^{\#z} \subset \mathbb{R}^d$. Note that under this formulation the quality estimator Λ corresponds to the description in Section II, but its domain now equals $E \times \mathcal{S}^\star \times \mathcal{P}(\Omega^\star)$. By expressing the model complexity in terms of the number of input parameters, multi-objective optimization of Equation 2 involving this formulation of the model quality estimator now also involves searching for a (preferably small) subset of the d input parameters which results in good modeling performance. This implies irrelevant features are excluded from models as they do not enhance accuracy, but result in higher complexity scores. The result of the optimization is a set of pareto-optimal solutions ranging from models including too few parameters (underfitting), to models including all parameters possibly overfitting the data.

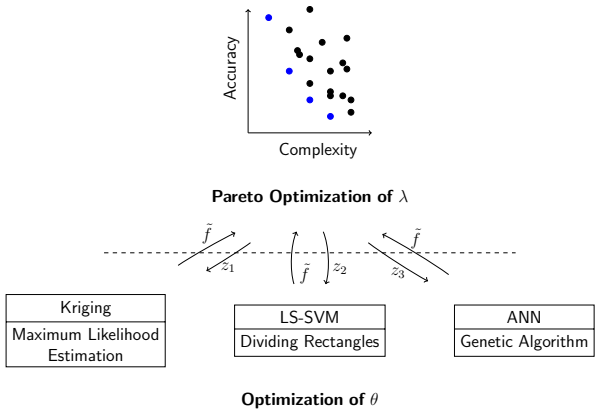


Fig. 2. Overview of the two-layered adaptive modeling process.

A. Finding pareto-optimal models

A genetic algorithm supporting multi-objective optimization such as NSGA-II [26] can be used to obtain P. Each individual of a population individual is represented by the vector $\lambda \in \mathbb{R}^{d+1}$ containing the surrogate model type, as well as the dimensions included by the model. To avoid varying vector dimensions, the dimensions in z are represented in λ by a binary vector of size d. In a second phase, the individuals of a population are trained on their assigned D_z, and automatic optimization of the hyperparameters θ is performed using a method which is usually specific for the model type t. Usually the method searches optimal θ in terms of enhancing model accuracy. The model quality estimator used during the optimization of θ can differ from Λ. As the model complexity remains constant during the hyperparameter optimization (the input variables do not change) so this objective is not considered during this phase. Only the model accuracy objective(s) are used. It makes sense to use the same accuracy objective(s), but this is not a strict requirement. For instance, model hyperparameters θ can be optimized with crossvalidation whereas the score on a validation set can be used as accuracy objective in the multi-objective optimization of λ.

After the models have been trained and optimal θ have been found, the model quality of the individuals are estimated with $\tilde{\Lambda}$. Fig. 2 presents an overview of this two-layered adaptive modeling step: the optimization of the hyperparameters θ does not occur at the level of the multi-objective optimization of λ. Because the hyperparameter space Θ is different for each surrogate model type, including the hyperparameters makes implementing a crossover and mutation operations less straightforward. Furthermore, adding θ to λ may significantly enlarge the search space and can result in good combinations of features resulting in poor model accuracy if the choice of θ is bad. When this individual is eliminated, it may take a few generations for this feature combination to re-appear. In addition, we find no evidence in literature that a different choice of model hyperparameters influences input parameter relevance (with respect to the complex system). Note that this selection approach differs from the island-based approach used earlier [17].

Unless the last generation has been generated and evaluated, the algorithm generates a new generation of models by performing a selection of the model individuals of the current generation to generate offspring. The selected models are given as input to a crossover and mutation operator. The crossover is defined to randomly pick one of the model types of the parents and takes a random subset of the union of the parameters of the parents. The mutation operator can randomly decide to change the model type, or randomly include or exclude a dimension. Some additional properties of these operations are discussed in the following sections. A schematic outline of the proposed adaptive modeling step for surrogate modeling with sequential design and automated feature selection is presented in Fig. 3.

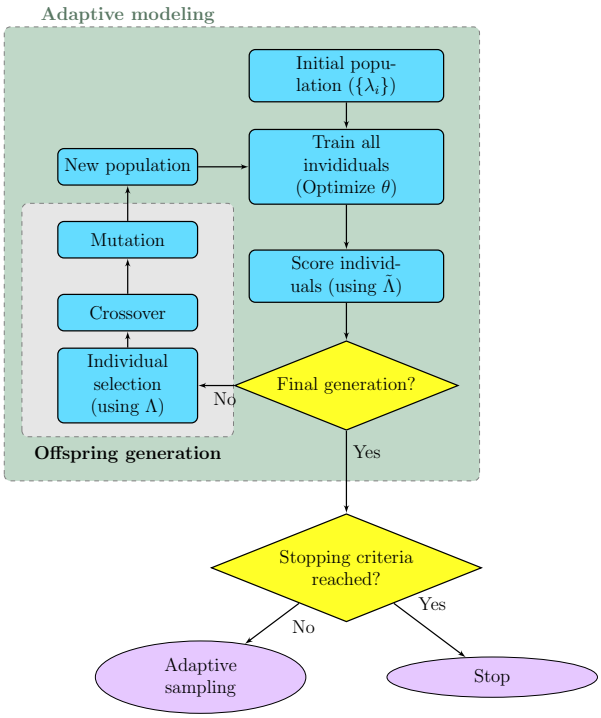


Fig. 3. Overview of the proposed adaptive modeling step with automated feature selection and sequential design.

B. Ensemble building

Ensemble models aggregate several $\tilde{f}$ and combine the outputs. Usually the models included in the ensemble each solve simpler subproblems, and the combination of all the subproblems results in an accurate solution to the initial problem. The most popular example of ensemble methods is Random Forests [27], which is essentially an ensemble of decision trees and can be used both for classification and regression problems.

With a small modification we can include ensemble individuals of the form

$$\tilde{f}^{*}_{\text{ens},\theta}(\boldsymbol{x}) = \sum_{i=1}^{l} \theta_i \tilde{f}_{\lambda_i,\bar{\theta}_i}(\boldsymbol{x}). \tag{4}$$

These linear combinations of surrogate models can be produced by the crossover operator with a small probability. Instead of choosing the model type of one of the parents (the standard behaviour of the crossover operator) the parents are grouped together. If one (or both) parents are ensembles, the child is an ensemble itself, including all models of the parents. For both cases, z is the union of the dimensions of the parents. The weight of the included models are the hyperparameters of ensemble models, whereas the hyperparameters of the includes models (denoted as $\bar{\theta}_i$) were the optimal hyperparameters found in the previous generation(s). When an ensemble model is chosen for mutation, the mutation operator randomly eliminates one of the contained models. If this implies the new ensemble contains only one model, the model itself becomes the child and the ensemble disappears.

As the individual models within the ensemble are not necessarily constructed using the same subspace, this means problems of higher dimensionality can be decomposed into model problems of a smaller dimensionality. This idea is closely related to HDMR [28], more specifically to recent work using surrogate models instead of polynomials [29].

C. Complexity objective

The complexity objective (denoted as $\mathcal{C}$) of $\tilde{\Lambda}$ penalizes models with many parameters to promote finding an optimal subset of input variables. It can therefore be defined as $\mathcal{C}(\tilde{f}_\lambda) = \#z$.

Although this definition results in feature selection, the complexity objective can also be used to further influence the model type-selection process. For example, an Ensemble of 2 SVM models, built on parameter i and j is less straightforward compared to an SVM of parameters i and j. Also the number of hyperparameters can play a role as this influences the associated cost of optimizing them. Building an SVM on a subset of 6 parameters is a lot cheaper as only the kernel bandwidth and the cost parameter need to be optimized. If instead, a Kriging model using the anisotropic Gaussian correlation function is to be built, 6 individual parameters need to be optimized. Including this information into $\mathcal{C}$ can steer the optimization

procedure to prefer easier models, and avoid many individual that require intensive training.

Table I shows the definition of $\mathcal{C}$ for the model types as used in the experiment of this paper. The definition was chosen manually by the authors, reflecting the time required to train and optimize each model type. Model types that are fast to train and do not result in a complex hyperparameter optimization problem are preferred whereas model types that do not offer these advantages will only survive selection if they significantly improve modeling accuracy. This formulation of the complexity also implies cheaper model types will influence the populations significantly in terms of feature relevance and can be considered as *worker models*: even though they may not be extremely accurate they will be preferred over complex models and cause removal of irrelevant features early during the process due to the selection. These feature subsets will also be used to determine the input parameters used to train more complex models through crossover and mutation. It is then up to these model types to improve the accuracy significantly to account for their expensive training.

D. Constraints to avoid infeasible models

Although $\mathcal{C}$ favours models with less complexity, this does not imply the crossover and mutation will never produce individuals higher complexity. For large datasets, this potentially results in very intensive model training procedures. For instance, optimizing the hyperparameters of Kriging becomes infeasible when many input parameters are included and an inproper correlation function is used. When D grows large during sequential design, some model types may also be inappropriate and require a lot of memory to complete the training phase. These types of constraints on the individuals must be included in the crossover and mutation functions to assure the generation of new populations is restricted to feasible individuals.

V. Experiment

A. Implementation and setup

The proposed methodology was implemented in the SUrrogate MOdeling (SUMO) Toolbox [30], a state-of-the-art research platform supporting both sequential design, one-shot design, and multi-objective hyperparameter optimization. Several surrogate model types (such as SVM [5], LS-SVM [7], ELM [8], [9], ANN, Kriging [3], Gaussian Procceses [31], Rational interpolation [10], splines etc.) and hyperparameter optimization algorithms (Particle Swarm Optimization [32], Genetic Algorithms, DIRECT [33], Simulated Annealing, Pattern Search [34] etc.) are available. The toolbox is designed according to a micro-kernel architecture, and is configured through a central XML file. Because of this, a lot of the building blocks required to implement the proposed strategy were available.

As test problem, the 30-dimensional function defined in [35] was chosen. It is defined as

$$\begin{aligned} f(\boldsymbol{x}) = &\quad \alpha \sum_{i=1}^{k_1} \left(x_i + \beta \sum_{i<j=2}^{k_1} x_i x_j \right) \text{ with} \\ \alpha = &\quad \sqrt{12} - \sqrt{0.1(k_1 - 1)} \\ \beta = &\quad 12\sqrt{0.1(k_1 - 1)}. \end{aligned}$$

For this experiment, $k_1 = 10$ which includes $x_1, ...x_{10}$, whereas all other parameters do not influence the response. Sequential design is used to find a surrogate model which approximates this function. The initial design consisted of 50 points, chosen randomly in the input space. More sophisticated schemes such as maximin Latin Hypercubes [36] are difficult to generate for this kind of high-dimensional space. Each iteration, the FLOLA-Voronoi [14] sampling algorithm was used to select 25 additional data points. This process continues until a total of 250 data points have been evaluated.

The proposed multi-objective optimization for automated selection of model type and optimal set of input parameters was used to build the intermediate models. In the experiment four different model types were included. The multi-objective quality estimator as introduced in Equation 3 was composed as follows: for the complexity objective values were assigned as defined in Table I. The scores were chosen manually, corresponding to the computational demands of training an individual of a model type (considering the time required to optimize its hyperparameters). To estimate the error (and assess the accuracy) of a model both during the hyperparameter optimization of each individual (with varying model type) as well in the multi-objective optimization, 5-fold crossvalidation was applied using the Root Relative Square Error error function:

$$\text{RRSE}(x, \tilde{x}) = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \tilde{x_i})^2}{\sum_{i=1}^{N} (x_i - \bar{x})^2}}. \tag{5}$$

For the GA performing the multi-objective optimization (λ), the population size was configured to 15 individuals. Each time the GA is called (i.e., once each iteration of the sequential design) 10 generations are generated, of which all individuals are trained and optimized. The crossover fraction was set to 0.7, and the probability of generating an ensemble in the crossover operation was 0.1. The initial population corresponds to the final generation obtained in the previous sequential design iteration, or is random when the modeling step is executed the first time.

Each model individual has its hyperparameters optimized using a specific algorithm based on its characteristics and parameters (θ). The LS-SVM and SVM models have two hyperparameters (kernel bandwidth and regularization parameter) which were optimized using the DIRECT [33] algorithm. Instead of traditional Artificial Neural Networks trained with backpropagation the ELM model type was included. Hyperparameter optimization of ANN would include the optimization of the network architecture with a genetic algorithm, which is a computationally demanding step especially in combination with crossvalidation. ELM are faster to train and consist of a single layer. As hyperparameters, the number of neurons in

TABLE I
OVERVIEW OF THE INCLUDED MODEL TYPES, THEIR HYPERPARAMETERS AND OPTIMIZATION STRATEGY

Surrogate model type	$\mathcal{C}$	Parameters θ	Optimizer
SVM	$\#z + 0.5$	Kernel bandwidth, Regularization constant	DIRECT
LS-SVM	$\#z$	Kernel bandwidth, Regularization constant	DIRECT
Kriging	$2\#z$	Kernel parameter	Maximum Likelihood Estimation
ELM	$2\#z$	Number of hidden neurons, Regularization constant, random number distribution width	Simulated Annealing
Ensemble	$\sum_i \mathcal{C}\left(\tilde{f}_i\right)$	Model weights	Pattern Search

the hidden layer, the regularization constant, and the range of the distribution used for the generation of the input weights were optimized using Simulated Annealing. The last model type included is Kriging, which uses Maximum Likelihood Estimation to determine its hyperparameters. Traditionally, Kriging uses an anisotropic correlation function which results in a hyperparameter for each dimension which can result in extremely expensive computations if a Kriging individual with many parameters is generated and must be trained. To avoid this issue, the isotropic Matern 3/2 correlation function was used, which has only a single hyperparameter regardless of the dimensionality. An overview of the included model types, their parameters and optimization strategies is given in Table I.

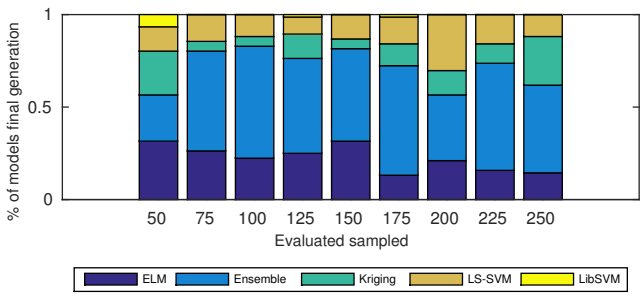


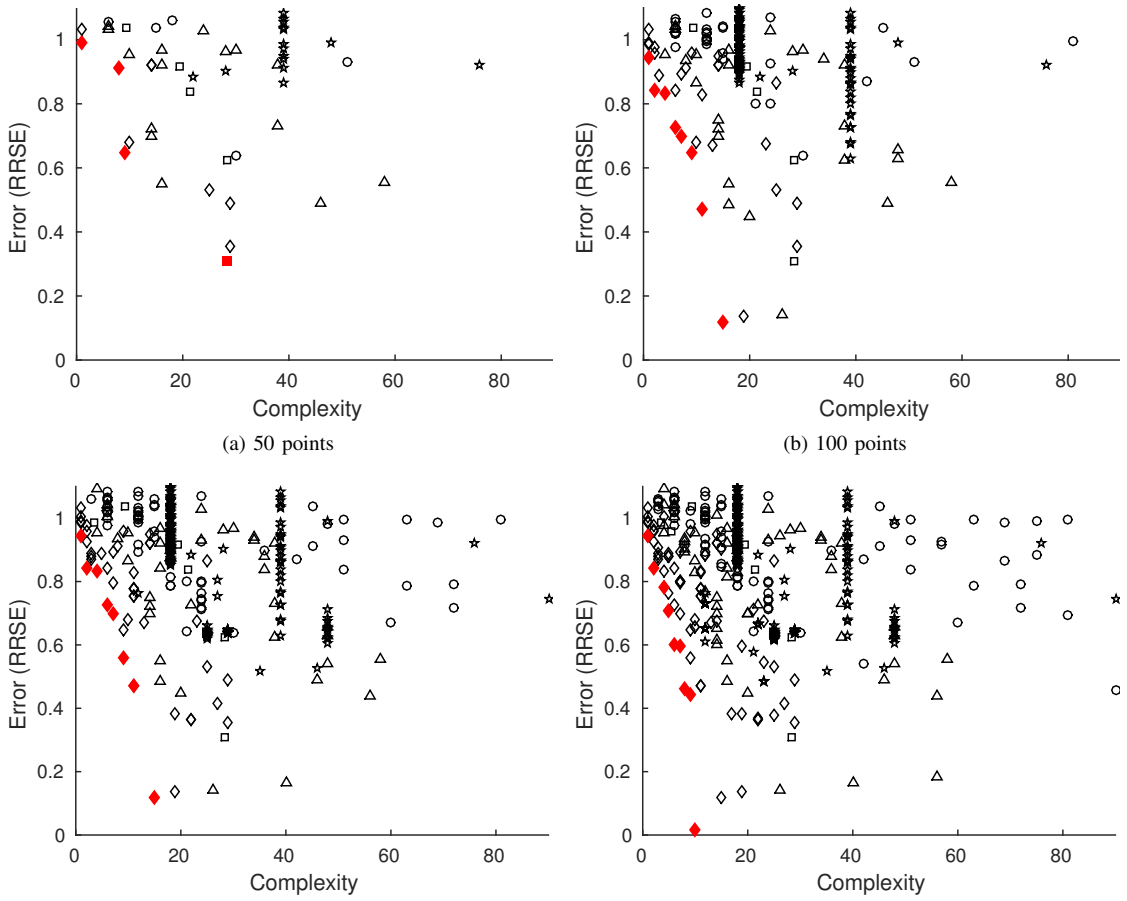
Fig. 5. The relative contribution of each model type to the final generation of the GA run, at each iteration of the sequential design.

B. Results and discussion

All resulting models that have been generated in any generation during the run are shown in Fig. 4, for four different iterations of the sequential design (with 50, 100, 150 and 200 evaluated samples available). Beyond 200 data points, the pareto front doesn't evolve anymore. The pareto front representing the trade-off between complexity and model error is shown in red, and the different model types are represented with different markers. It is clear that for this problem, the pareto front found after the third run of the adaptive modeling algorithm is already very satisfying. Clearly, the best scoring LS-SVM model has already identified most of the relevant parameters. When over 150 data points have been evaluated the adaptive modeling algorithm finds an LS-SVM which is even better, obtaining an almost perfect accuracy with exactly 10 input parameters. Models which include less parameters generally obtain scores which are worse as they are underfitting the problem.

From Fig. 4, it is clear LS-SVM models are performing very well for this experiment, whereas SVM models seem to be completely ignored. This observation is confirmed in Fig. 5 which shows the relative share of the final generation of each model type, for each run of the adaptive modeling step. Generations are mostly made up of several ensembles which seem to be generated often, as most of them are eliminated quickly due to their higher complexity. The ELM models are included mostly in the beginning (given their reasonable accuracies at that stage) but as more data points become available they do not seem to benefit as much as other methods and it can be observed they are selected less for generation of offspring. The Kriging models however seem to be less relevant in the beginning, but as more data points become available they seem to improve their scores. This is also clear in Fig. 4d: Kriging models generally obtain better scores compared to the ELM models. However, they do not show up in the pareto front, as LS-SVM models with similar error tend to have better complexity scores. This is also the motivation behind the elimination of the SVM model type: manual analysis of the output reveals most SVM models obtained scores equal to scores obtained with LS-SVM models. However due to the complexity penalty, the LS-SVM models were usually preferred.

In general, the proposed method proves to be successful at identification of relevant input parameters which opens up a lot of possibilities for surrogate modeling of high-dimensional problems. In this experiment most model types are able to approximate the response surface very well and the model type selection was mostly driven by our specification of the model complexity defined mostly in terms of the training time. This however helps us to tackle the biggest disadvantage of our proposed method: the total training time. As many models need to be trained (including hyperparameter optimization), a preference towards models types which perform this task faster is justified. It can also be observed in Fig. 4d that not many models have obtained a satisfying score. Analysis revealed that many models trained with 8 to 15 parameters were trained on the wrong subset of parameters. Allowing more generations in



(a) 50 points (b) 100 points

(c) 150 points (d) 200 points

Fig. 4. All trained models as part of the adaptive modeling step, at four different iterations of the sequential design process. Surrogate model types are denoted by ◊ for LS-SVM, □ for SVM, △ for Kriging, ○ for ELM and ⋆ for Ensemble models. The pareto front P has been marked in red, and its markers are filled.

the adaptive modeling step overcomes this problem as during the generation of extra generations the right variable subset becomes more popular and more models will obtain good accuracies. However, given the size of the search space and the total amount of models built on quite a small set of data, it is very satisfying to observe that our approach has managed to find identify the proper set of variables in reasonable time (the run took 3 hours in total to complete). Whether or not this worth the effort depends on the type of application. If obtaining a single data point from the simulator is a very lengthy process and the input space is high-dimensional, our proposed methodology potentially saves a lot of computing time.

VI. Conclusion

This paper contributes a new approach to the adaptive modeling step of the surrogate modeling process with sequential design. It extends earlier heterogeneous model selection approaches [17] with automated feature selection to identify relevant input parameters. This is a significant improvement for surrogate modeling of applications with a high-dimensional input space, as parameters with low significance can be excluded resulting in models that are easier to interpret. Furthermore, a significantly smaller input space needs to be approximated. Application of our methodology to a 30-dimensional problem shows that although the total search space is big, our method using multi-objective optimization can discover an optimal set of input parameters relatively quick.

Future work focusses on enhancing the crossover and mutation operators to find optimal subsets of input variables even faster, and get more out of the available model types. Also the construction of ensembles can be improved with more recent developments. In addition, the adaptive sampling schemes should avoid collapsing samples due to the dimensionality projection. Also, information from the best models obtained so far can be included into the sampling to focus stronger on relevant input parameters: this should result in a better interplay between the modeling and sampling algorithms. Also recent developments on tuning experiments for the NSGA-II algorithm can be included.

Acknowledgment

This research has (partially) been funded by the Inter university Attraction Poles Programme BESTCOM initiated by the Belgian Science Policy Office. Ivo Couckuyt is a post-doctoral research fellow of FWO-Vlaanderen.

REFERENCES

[1] K. Goethals, I. Couckuyt, T. Dhaene, and A. Janssens, "Sensitivity of night cooling performance to room/system design : surrogate models based on CFD," *Building and Environment*, vol. 58, pp. 23–36, 2012.

[2] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 989–993, 1994.

[3] I. Couckuyt, T. Dhaene, and P. Demeester, "ooDACE Toolbox: A Flexible Object-Oriented Kriging Implementation," *Journal of Machine Learning Research*, vol. 15, pp. 3183–3186, 2014.

[4] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992, pp. 144–152.

[5] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[6] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in large margin classifiers*. Citeseer, 1999.

[7] J. A. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, J. Suykens, and T. Van Gestel, *Least squares support vector machines*. World Scientific, 2002, vol. 4.

[8] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.

[9] G.-B. Huang, D. H. Wang, and Y. Lan, "Extreme learning machines: a survey," *International Journal of Machine Learning and Cybernetics*, vol. 2, no. 2, pp. 107–122, 2011.

[10] D. Deschrijver, T. Dhaene, and D. De Zutter, "Robust parametric macromodeling using multivariate orthonormal vector fitting," *IEEE Trans. Microw. Theory Tech*, vol. 56, no. 7, pp. 1661–1667, 2008.

[11] T. Simpson, D. Lin, and W. Chen, "Sampling Strategies for Computer Experiments: Design and Analysis," *International Journal of Reliability and Application*, vol. 2, no. 3, pp. 209–240, 2002.

[12] D. Gorissen, "Grid-enabled adaptive surrogate modeling for computer aided engineering," 2010.

[13] K. Crombecq, D. Gorissen, D. Deschrijver, and T. Dhaene, "A Novel Hybrid Sequential Design Strategy for Global Surrogate Modelling of Computer Experiments," *SIAM Journal of Scientific Computing*, vol. 33, no. 4, pp. 1948–1974, 2010.

[14] J. van der Herten, I. Couckuyt, D. Deschrijver, and T. Dhaene, "A Fuzzy Hybrid Sequential Design Strategy for Global Surrogate Modeling of High-Dimensional Computer Experiments," *SIAM Journal on Scientific Computing*, vol. 37, no. 2, pp. A1020–A1039, 2015.

[15] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient Global Optimization of Expensive Black-Box Functions," *J. of Global Optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[16] I. Couckuyt, D. Deschrijver, and T. Dhaene, "Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization," *Journal of Global Optimization*, vol. 60, no. 3, pp. 575–594, 2014.

[17] D. Gorissen, T. Dhaene, and F. De Turck, "Evolutionary Model Type Selection for Global Surrogate Modeling," *Journal of Machine Learning Research*, vol. 10, pp. 2039–2078, 2009.

[18] A. Rosales-Pérez, J. A. Gonzalez, C. A. C. Coello, H. J. Escalante, and C. A. Reyes-Garcia, "Multi-objective model type selection," *Neurocomputing*, vol. 146, pp. 83–94, 2014.

[19] H. Akaike, "A new look at the statistical model identification," *Automatic Control, IEEE Transactions on*, vol. 19, no. 6, pp. 716–723, 1974.

[20] D. Gorissen, I. Couckuyt, E. Laermans, and T. Dhaene, "Pareto-based multi-output metamodeling with active learning," in *Proceedings of the 11th International Conference on Engineering Applications of Neural Networks (EANN 2009), London, England*, 2009.

[21] K. Ritter, G. W. Wasilkowski, and H. Woźniakowski, "On multivariate integration for stochastic processes," in *Numerical Integration IV*. Springer, 1993, pp. 331–347.

[22] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *Machine Learning*, vol. 3, pp. 1157–1182, 2003.

[23] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *bioinformatics*, vol. 23, no. 19, pp. 2507–2517, 2007.

[24] S. Peng, Q. Xu, X. B. Ling, X. Peng, W. Du, and L. Chen, "Molecular classification of cancer types from microarray data using the combination of genetic algorithms and support vector machines," *FEBS letters*, vol. 555, no. 2, pp. 358–362, 2003.

[25] T. K. Paul and H. Iba, "Selection of the most useful subset of genes for gene expression-based classification," in *Evolutionary Computation, 2004. CEC2004. Congress on*, vol. 2. IEEE, 2004, pp. 2076–2083.

[26] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[27] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[28] G. Li, J. Hu, S.-W. Wang, P. G. Georgopoulos, J. Schoendorf, and H. Rabitz, "Random sampling-high dimensional model representation (RS-HDMR) and orthogonality of its different order component functions," *The Journal of Physical Chemistry A*, vol. 110, no. 7, pp. 2474–2485, 2006.

[29] S. Ulaganathan, I. Couckuyt, T. Dhaene, J. Degroote, and E. Laermans, "High dimensional Kriging metamodelling utilising gradient information," *Applied Mathematical Modelling*, pp. –, 2015.

[30] D. Gorissen, K. Crombecq, I. Couckuyt, P. Demeester, and T. Dhaene, "A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design," *Journal of Machine Learning Research*, vol. 11, pp. 2051–2055, 2010, available at http://sumo.intec.ugent.be.

[31] C. E. Rasmussen and H. Nickisch, "Gaussian processes for machine learning (GPML) toolbox," *The Journal of Machine Learning Research*, vol. 11, pp. 3011–3015, 2010.

[32] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*. Springer, 2010, pp. 760–766.

[33] D. R. Jones, C. D. Perttunen, and B. E. Stuckman, "Lipschitzian optimization without the Lipschitz constant," *Journal of Optimization Theory and Applications*, vol. 79, no. 1, pp. 157–181, 1993.

[34] R. Hooke and T. A. Jeeves, ""Direct Search"Solution of Numerical and Statistical Problems," *Journal of the ACM (JACM)*, vol. 8, no. 2, pp. 212–229, 1961.

[35] M. D. Morris, L. M. Moore, and M. D. McKay, "Sampling plans based on balanced incomplete block designs for evaluating the importance of computer model inputs," *Journal of Statistical Planning and Inference*, vol. 136, no. 9, pp. 3203–3220, 2006.

[36] E. Dam, B. van Husslage, D. den Hertog, and J. Melissen, "Maximin Latin hypercube designs in two dimensions," *Operations Research*, vol. 55, no. 1, pp. 158–169, 2007.